K.I.E.T GROUP OF INSTITUTION

PROJECT TITLE:

STUDENT PERFORMANCE REPORT

SUBMITTED BY - HIMANI BHATT

COURSE NAME – INTRODUCTION TO AI

COURSE CODE – AI101B

DATE – 11-03-2025

INTRODUCTION

In today's data-driven world, artificial intelligence (AI) plays a significant role in analyzing and predicting student performance. This project, **Student Performance Prediction**, aims to utilize machine learning techniques to forecast students' final exam scores based on their study hours and previous academic performance. By leveraging historical data, we can identify patterns that help students and educators make informed decisions to improve academic outcomes.

Project Overview

The dataset used in this project consists of student records, including:

- **Study Hours**: The number of hours a student spends studying.
- **Previous Scores**: The student's past academic performance.
- **Final Exam Score**: The target variable that we aim to predict.

Objective

The primary objective of this project is to build a predictive model using **linear regression**, which helps estimate students' final exam scores based on their study habits and previous performance. This information can be used to:

- Identify students who may need additional support.
- Provide insights for educators to optimize study plans.
- Enable students to understand the impact of study habits on their performance.

METHODOLOGY

1. Data Collection
    - Gather data on students, including their study hours, previous scores, and final exam scores.
    - Store this data in a CSV file for easy access.
2. Data Preprocessing
    - Load the dataset using Python libraries like Pandas.
    - Check for missing or inconsistent values and clean the data.
    - Split the dataset into training and testing sets (80% for training, 20% for testing).
3. Model Selection & Training
    - Use a simple Linear Regression model to find patterns in the data.
    - Train the model using the training dataset, mapping study hours and previous scores to final exam scores.
4. Model Evaluation
    - Test the model's predictions on the test dataset.
    - Measure accuracy using error metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).
5. Visualization & Analysis
    - Plot graphs such as scatter plots, bar graphs, and line charts to visualize trends.
    - Compare actual vs predicted scores to understand the model's performance.

CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error


# Create and save dataset as CSV
csv_filename = "student_performance.csv"
data_dict = {
    "StudentID": list(range(1, 21)),
    "StudyHours": [8.777482, 9.161915, 3.278010, 4.500247, 2.264931, 5.178765, 5.195977, 7.551904, 1.940879, 3.315756,
                   4.599099, 2.471536, 6.708061, 6.692461, 9.654504, 6.392792, 4.655716, 6.339880, 6.195849, 7.819022],
    "PreviousScores": [75, 55, 77, 60, 72, 87, 45, 68, 73, 78, 99, 68, 63, 44, 85, 86, 57, 69, 52, 90],
    "FinalExamScore": [64, 82, 70, 60, 60, 81, 85, 57, 65, 68, 81, 96, 85, 93, 52, 43, 99, 42, 76, 79]
}
data = pd.DataFrame(data_dict)
data.to_csv(csv_filename, index=False)
print(f"CSV file '{csv_filename}' has been created successfully.")


# Load dataset
data = pd.read_csv(csv_filename)
```

```
# Display first few rows
print("Dataset Preview:")
print(data.head())

# Define independent variables (features) and dependent variable (target)
X = data[['StudyHours', 'PreviousScores']]
y = data['FinalExamScore']

# Split data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize and train the model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on test data
y_pred = model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print("\nModel Performance:")
```

```
print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")

# Plot actual vs predicted values
plt.scatter(y_test, y_pred, color='blue')
plt.xlabel("Actual Final Exam Score")
plt.ylabel("Predicted Final Exam Score")
plt.title("Actual vs Predicted Final Exam Scores")
plt.show()

# Display the model's coefficients and intercept
print("\nModel Coefficients:")
print(f"StudyHours Coefficient: {model.coef_[0]:.2f}")
print(f"PreviousScores Coefficient: {model.coef_[1]:.2f}")
print(f"Intercept: {model.intercept_:.2f}")

# Additional Visualizations
plt.figure(figsize=(12, 6))

# Bar graph for Study Hours vs Final Exam Score
plt.subplot(1, 2, 1)
plt.bar(data['StudentID'], data['StudyHours'], color='skyblue', label='Study Hours')
plt.bar(data['StudentID'], data['FinalExamScore'], color='orange', alpha=0.7,
label='Final Exam Score')
plt.xlabel("Student ID")
```

```
plt.ylabel("Values")
plt.title("Study Hours & Final Exam Score Comparison")
plt.legend()

# Line graph for Previous Scores vs Final Exam Score
plt.subplot(1, 2, 2)
plt.plot(data['StudentID'], data['PreviousScores'], marker='o', linestyle='-', color='red', label='Previous Scores')
plt.plot(data['StudentID'], data['FinalExamScore'], marker='s', linestyle='--', color='green', label='Final Exam Score')
plt.xlabel("Student ID")
plt.ylabel("Scores")
plt.title("Previous Scores vs Final Exam Score")
plt.legend()

plt.tight_layout()
plt.show()
```

OUTPUT

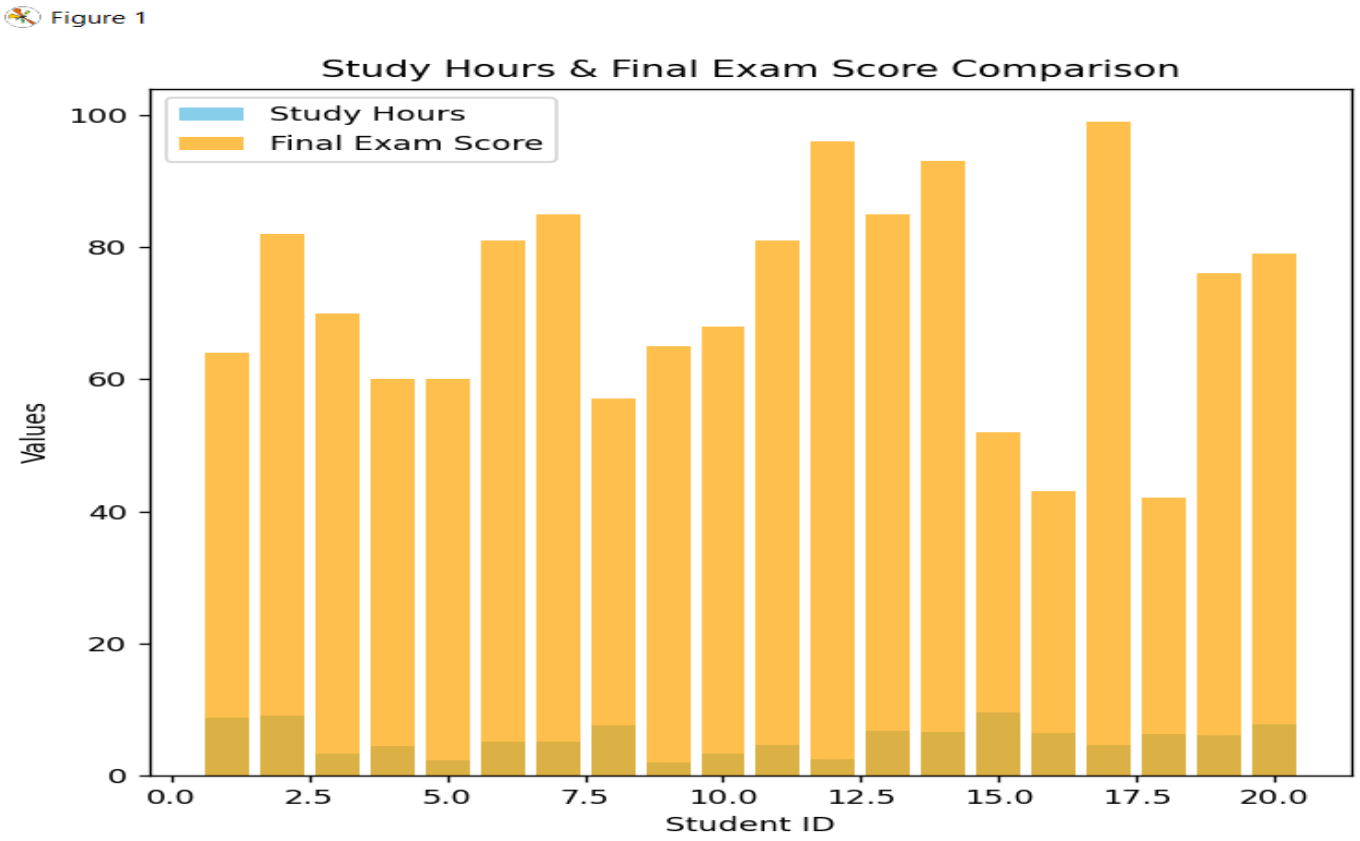
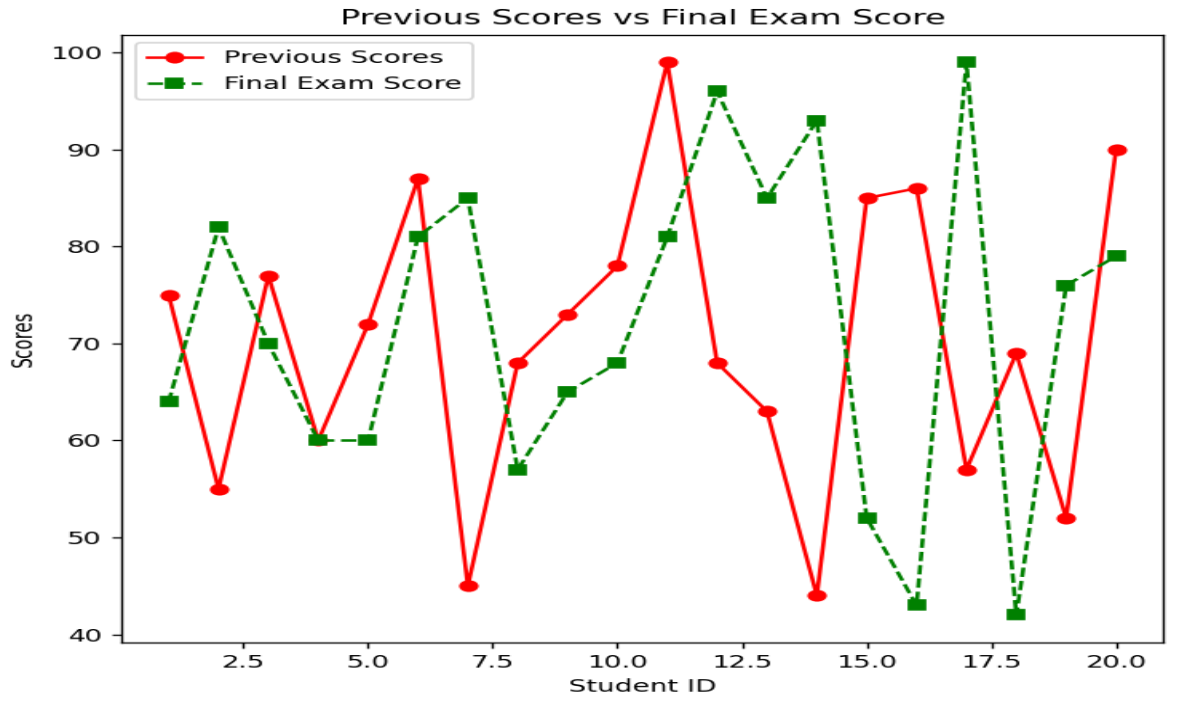


Figure 1
Study Hours & Final Exam Score Comparison
Study Hours
Final Exam Score
Values
Student ID
0
20
40
60
80
100
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Previous Scores vs Final Exam Score
Previous Scores
Final Exam Score
Scores
Student ID
40
50
60
70
80
90
100
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0